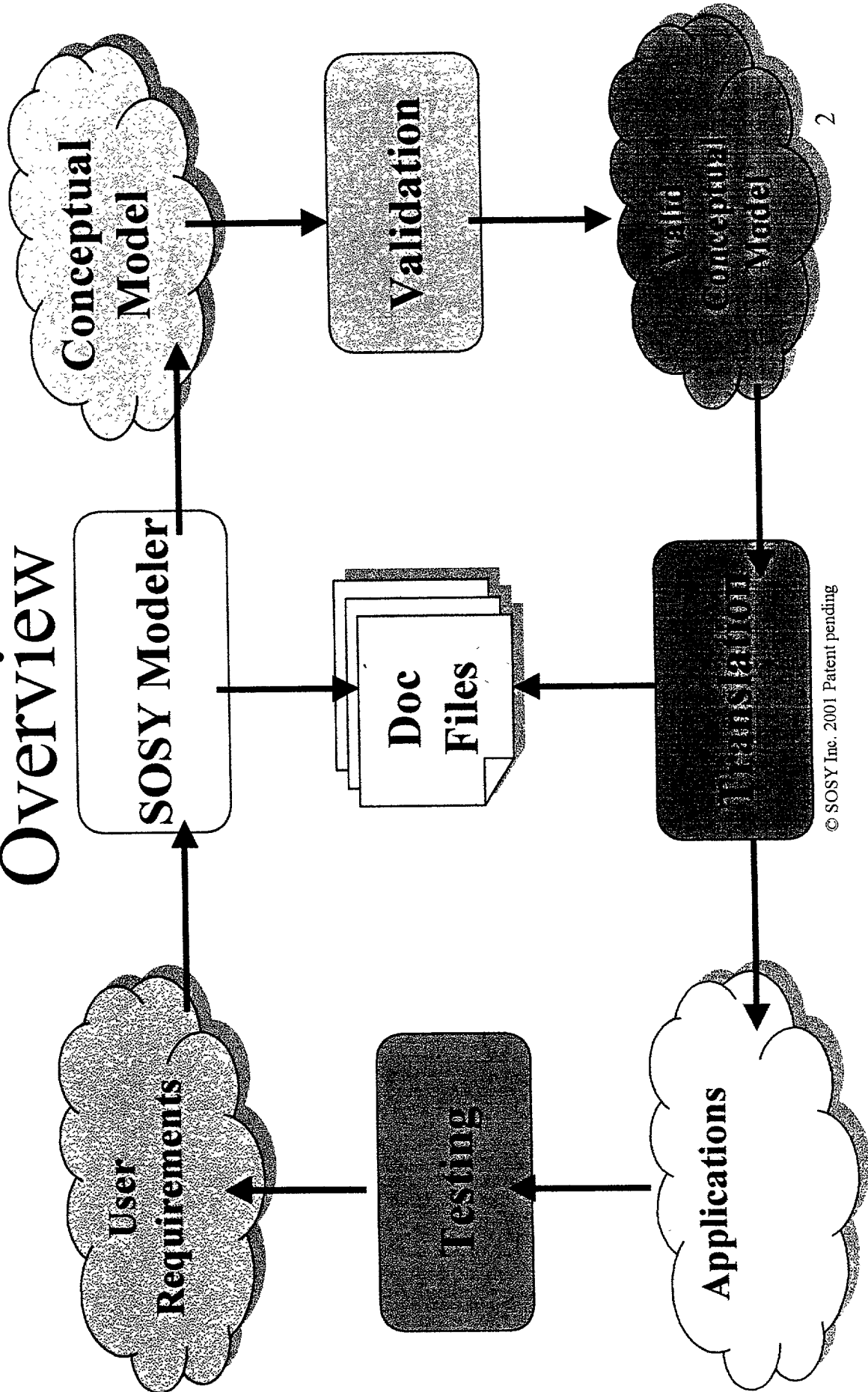


Summary

- Model
- Validation
- Deployment
- Persistence
- Business Logic
- User Interface

Overview



Conceptual Modeling Phase

CARE Technologies, S.A.

Index

- Intro
- Overview
- Phase 0. Requirements elicitation.
- Phase 1. Classes identification.
- Phase 2. Relationships between classes.
- Phase 3. Filling classes' details.

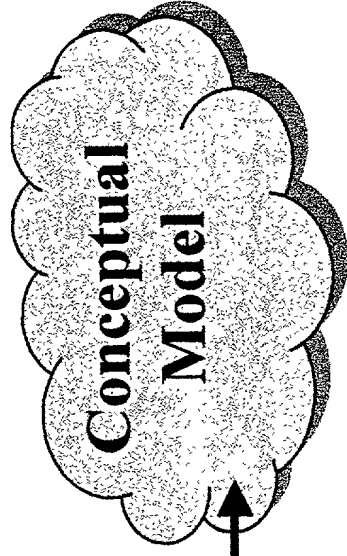
Index

- Phase 4. Express evaluations.
- Phase 5. Agent relationships.
- Phase 6. State Transition Diagram.
- Phase 7. Presentation Model.

Intro

- Conceptual Modeling Phase is a process of systematically & precisely description of the system to build, using:
 - Graphical UML compliant diagrams.
 - Constrains and semantics in a formal non-ambiguous language.
 - This phase is assisted by an integrated Modeler tool.

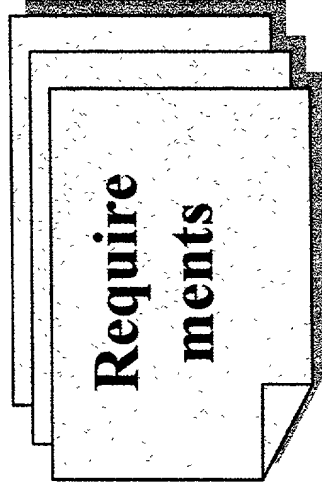
Overview



Conceptual Model

- Classes
- Relationships
- Attributes
- Services
- ...

Expressed in a non-ambiguous language.



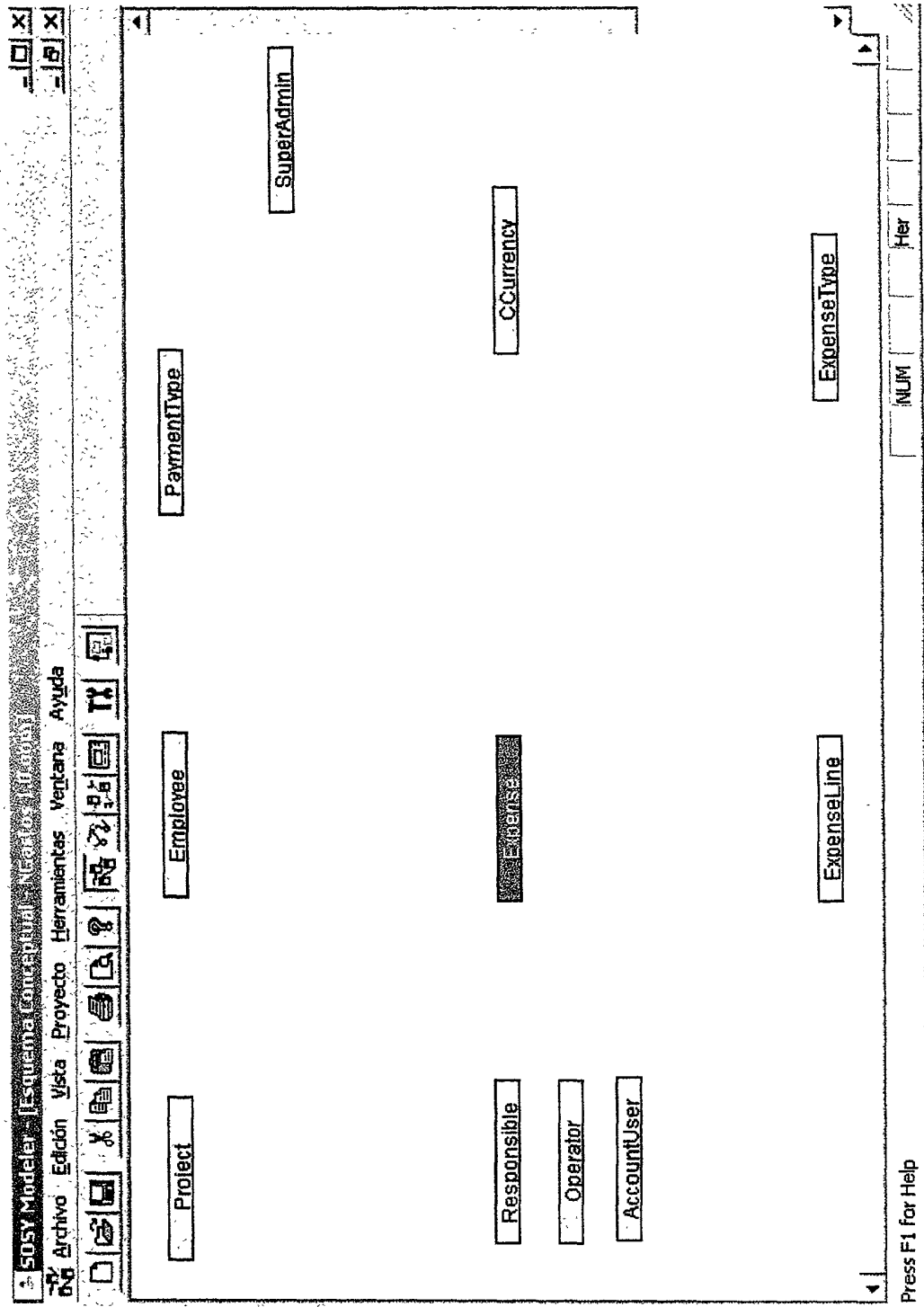
Requirements

- Specifications
- Documents
- Interviews
- Reports
- Other info. sources

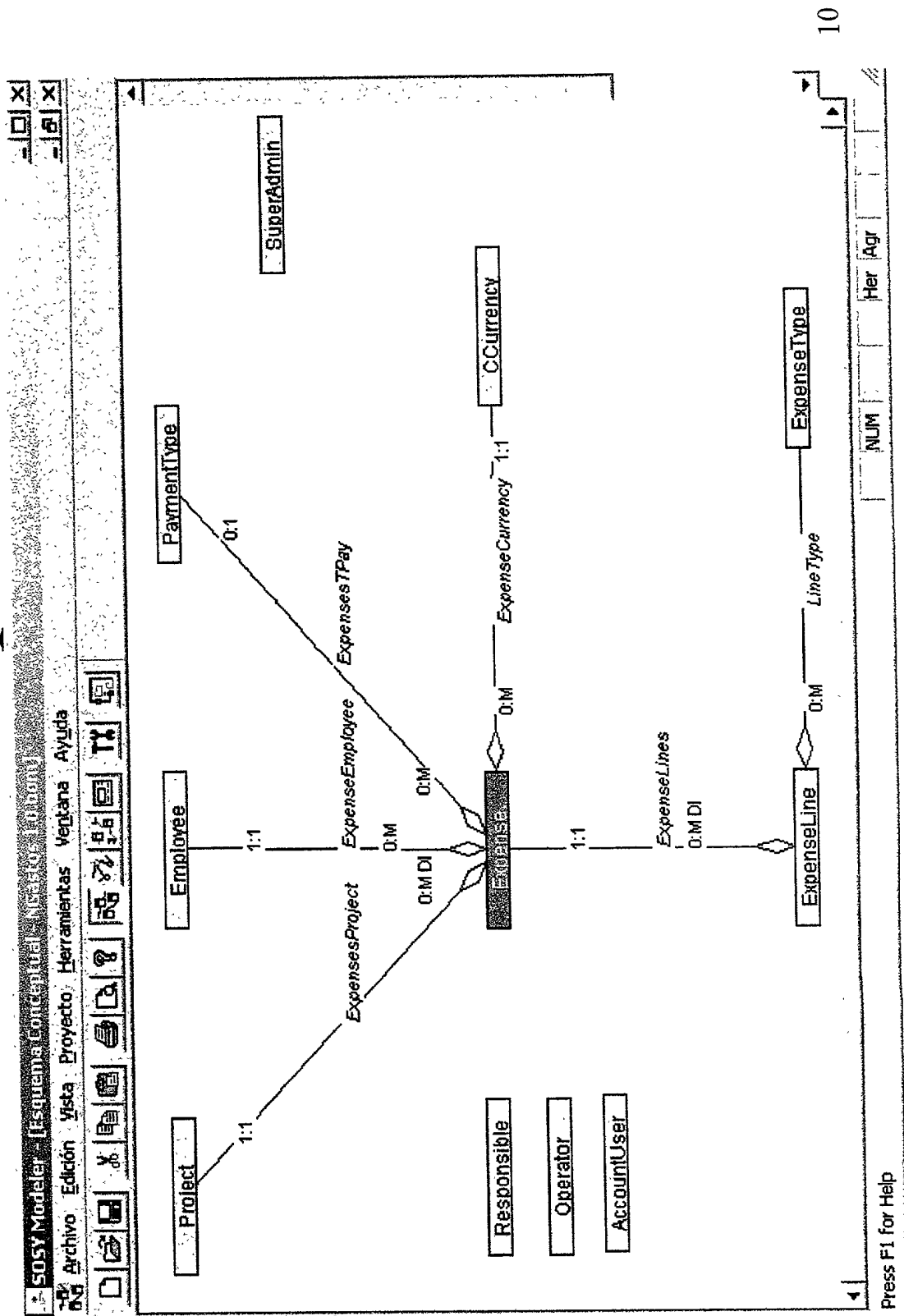
Phase 0. Requirement elicitation.

- Gathering the system requirements.
 - By meetings & interviews with customers, experts and final users.
 - By collecting reports, or documents expressing the system how-to and using tools.
 - Obtaining a coherent set of information as input to the next phase.

Phase 1. Classes identification.



Phase 2. Relationships between classes.



Atributos									
Nombre	Tipo atributo	Tipo dato	Id	Tamaño	Valor defecto	Pedir al crear	Nulos	Añadir	
PresentDate	Constante	Date			today()	Sí	No	<input type="button" value="Modificar"/>	
Status	Variable	Int			0	Sí	No	<input type="button" value="Borrar"/>	
Cause	Variable	String		255			No		
AuthoDate	Variable	Date			NULL	No	Sí		
AuthoComments	Variable	String		255	NULL	No	Sí		
PaymentDate	Variable	Date			NULL	No	Sí		
PayComments	Variable	String		255		No	Sí		
TotExpenses	Derivado	Real							
TotExpensesCur	Derivado	Real			0	Sí	No		
Advances	Variable	Real							
AdvancesCur	Derivado	Real				No	Sí		
Exchange	Variable	Real							
Balance	Derivado	Real							
BalanceCur	Derivado	Real							

Tipo Atributo: Tipo Dato:

Alias:

Observaciones:

☐ Información Temporal

Phase 3. Filling classes' details.

Clase

Arbitros | Servicios | Derivaciones | Restricciones | Agentes | Transacciones | Relaciones | Generalidades

Atributo

Balance

Fórmulas de Derivación

Condición

Fórmula

Total Expenses - Advances

Añadir

Modificar

Borrar

Condición

Fórmula

Observaciones

Clase:

Expense

Aceptar

Cancelar

[illegible]13

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99
0	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99

14

[illegible]15

16

Phase 5. Agent relationships.

Clase

Atributos

Servicios

Derivaciones

Restricciones

Agentes

Transacciones

Relaciones

Generalidades

Clase servidora:

Expense

Servicios:

Servicio

newexpense

modify

close

authorize

rejectautho

DELETEALL

Clase AccountUser agente de:

Servicio

Expense.approve

Expense.rejectpayment

Expense.TPAY

Temporalidad

No

No

No

Hacer temporal

No hacer temporal

Clase AccountUser tiene visibilidad sobre:

Atributos

Expense.id

Expense

Expense.PresentDate

Expense.Status

Expense.PaymentDate

Expense.PayComments

Expense.AuthoDate

Expense.TotExpenses

Expense.TotExpensesCur

Expense.Advances

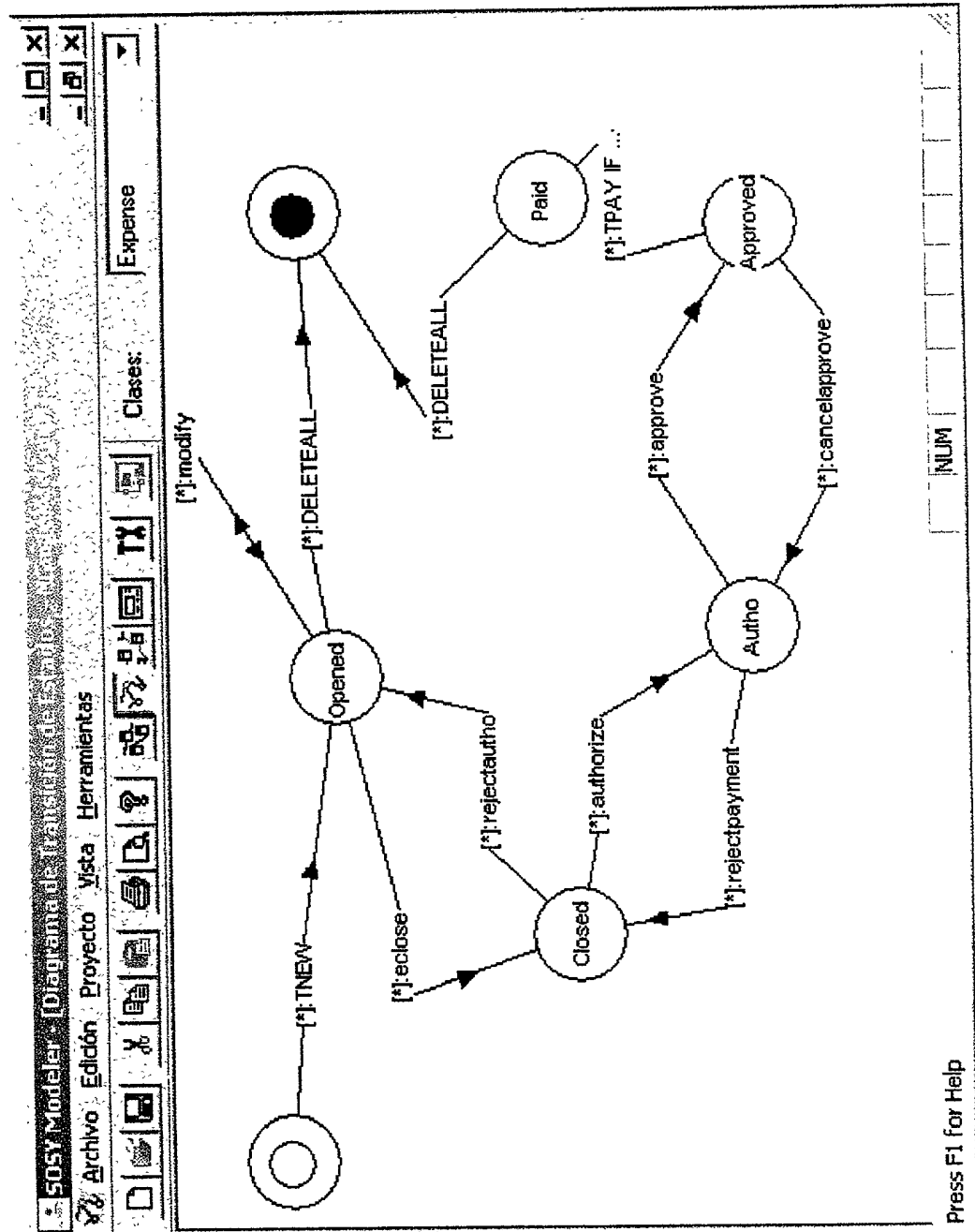
Clase:

AccountUser

Aceptar

Cancelar

Phase 6. State Transition Diagram.



Phase 6. STD Preconditions

Transición

Origen :
Approved

Destino :
Paid

Acceptar

Cancelar

Detalles

Agentes:
AccountUser
SuperAdmin

Servicio:
TPAY

Precondición:
Balance > 0 OR ps_ReturnAdvance = TRUE

Condición de control:

Mensaje en caso de Error:

Check the advanced money excess

Phase 7. Presentation Model.

Conjunto de Visualización

Nombre: CV_Expense

Limpia

Borra

Atributos a visualizar:

Atributo	Tipo dato
Project.ProjectName	String
Employee.EmpName	String
Employee.EmpSur...	String
Status	Int
AuthoDate	Date
PaymentDate	Date
TotExpenses	Real
Balance	Real

<< Añadir

Eliminar >>

Subir

Bajar

Agregar

Atributos:

Atributo	Tipo dato
Cause	String
AuthoDate	Date
AuthoComments	String
PaymentDate	Date
PayComments	String
TotExpenses	Real
TotExpensesCur	Real
Advances	Real
AdvancesCur	Real
Exchange	Real
Balance	Real
BalanceCur	Real

Clase: Expense

Aceptar

Cancelar

Phase 7. Presentation Model.

Filtro

flt_Expense

Alias:

Expense Reports

Limpiar

Borrar

Fórmula:

Project = vf_Project AND Employee = vf_Employee AND PresentDate >= vf_DateIniIssue AND PresentDate <= vf_DateEndIssue AND AuthoDate >= vf_DateIniApp AND AuthoDate <= vf_DateEndApp AND PaymentDate >= vf_DateIniPay AND PaymentDate <= vf_DateEndPay AND

<< Variable

Observ:

Variables

Nombre	Alias	Tipo dato	Tipo estilo	Estilo	Nueva	Modificar	Borrar
vf_Project	Project	Project	Sel Población	Sel Población			
vf_Employee	Employee	Employee					
vf_DateIniIssue	Initial Issuing Date	Date					
vf_DateEndIssue	Final Issuing Date	Date					
vf_DateIniApp	Initial Approving D...	Date					

Tipo

☒ Simple

☐ Objeto-valuado

Nombre:

Alias:

Tipo de dato:

Estilo de introd.:

Estilo de selección:

Aceptar

Cancelar

Clase:

Expense

Conceptual Model Validation

CARE Technologies, S.A.

Index

- Intro
- Overview
- Validation Degrees
 - Partial Validation
 - Total Validation

Index

- Validation Types
 - Elements of the Conceptual Model
 - Formulas of the Conceptual Model (Syntax)
- Validation Trees
 - Nodes
 - Leaves
- Example

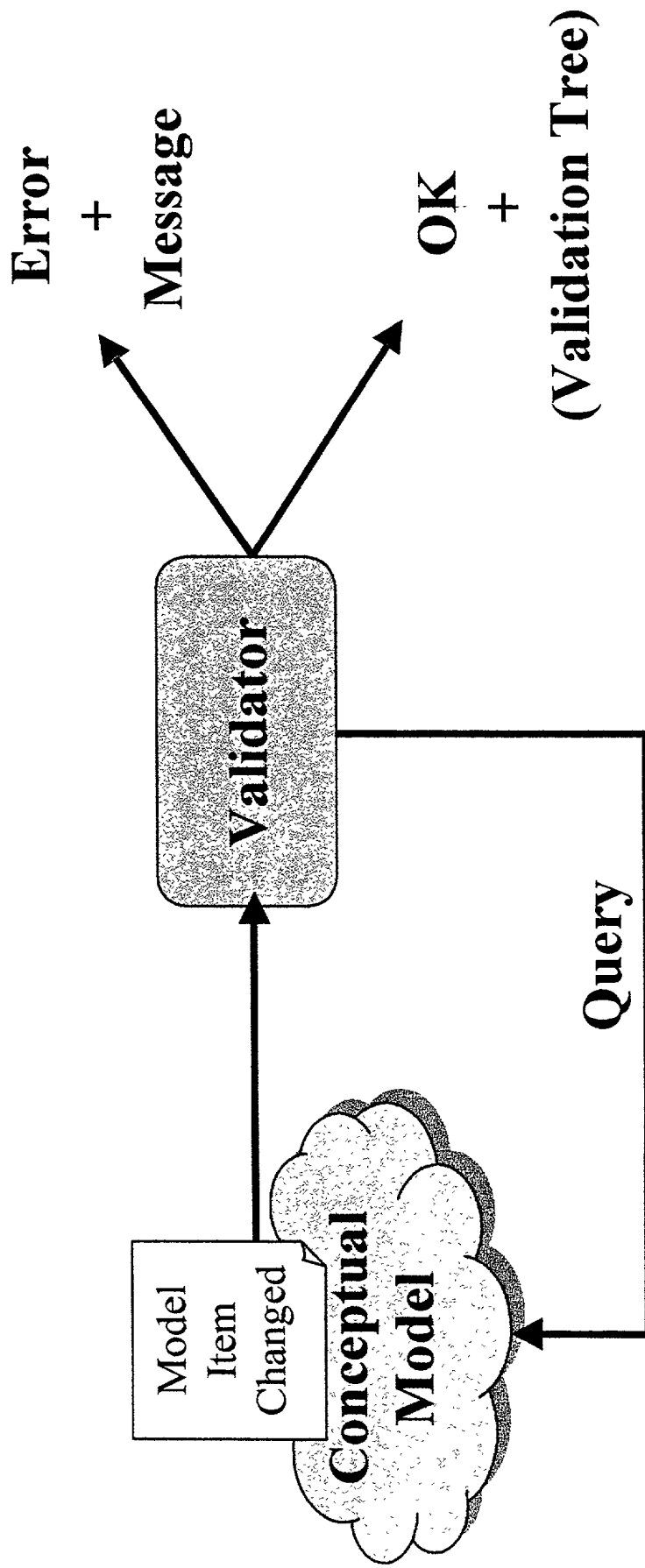
Intro

- Conceptual Model Validation is the process by which a conceptual model or a modification of it is proven to be valid:
 - Correct
 - Non Ambiguous
 - Non Contradictory
 - Complete
 - Every concept is fully specified
- Validation process checks the representation of requirements in Formal Specification Language to be valid

Validation Degrees

- Partial Validation
 - That of a single element of the Conceptual Model.
 - Happens whenever an element is added, modified or deleted.

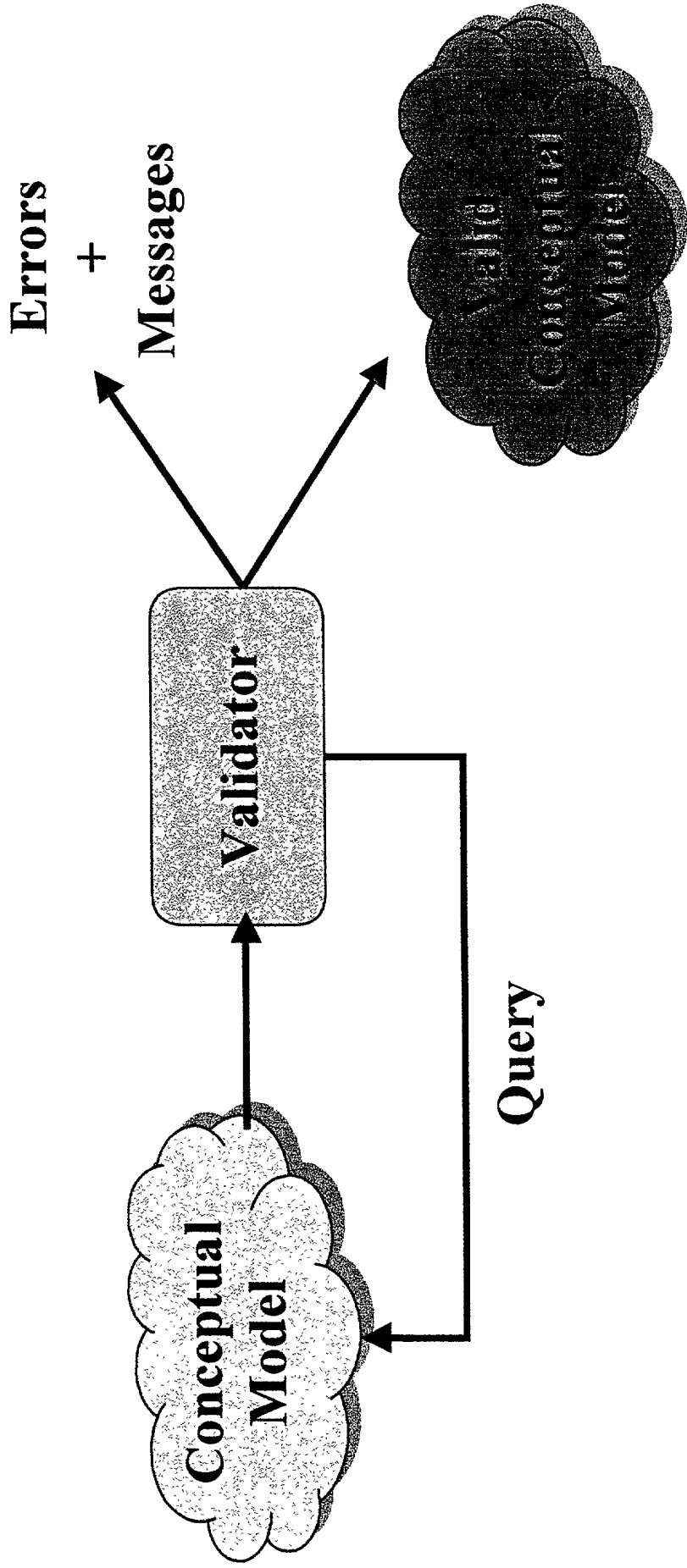
Partial Validation Overview



Validation Degrees

- Total Validation
 - That of the whole Conceptual Model.
 - Happens by request.
 - Must happen prior to any translation process.
 - Takes advantage of partial validations already performed.

Total Validation Overview



Total Validation Example

xl

Validación del modelo

Verificaciones:

Aviso :El parametro 'pt_p_thisExpense' de la transacción 'TPAY' de la clase 'Expense' no se utiliza

Aviso :El parametro 'ps_ReturnAdvance' de la transacción 'TPAY' de la clase 'Expense' no se utiliza

Analizando la clase Employee

Analizando la clase ExpenseLine

Analizando la clase ExpenseType

Analizando la clase PaymentType

Analizando la clase Responsible

Aviso :El atributo 'ResName' de la clase 'Responsible' no tiene evaluaciones definidas

Aviso :El atributo 'ResSurname' de la clase 'Responsible' no tiene evaluaciones definidas

Analizando la clase Operator

Aviso :El atributo 'OpName' de la clase 'Operator' no tiene evaluaciones definidas

Aviso :El atributo 'OpSurname' de la clase 'Operator' no tiene evaluaciones definidas

Analizando la clase AccountUser

Analizando la clase CCurrency

Analizando la clase SuperAdmin

Validación de transacciones globales...

=====

Resultados : 0 error(es), 7 aviso(s).

Validar

Cancelar

Imprimir!

Volcar a Fichero

Volcar XML

Cerrar

Validation Types

- Elements of the Conceptual Model
 - Ensure the properties of an element (except formulas) are correct and complete.
 - Conditions that must hold depend on the type of element and the property being validated.
- Examples:
 - Class Name is unique in a Conceptual Model.
 - Attribute Name is unique in its Class (but not in a Conceptual Model)

Validation Types

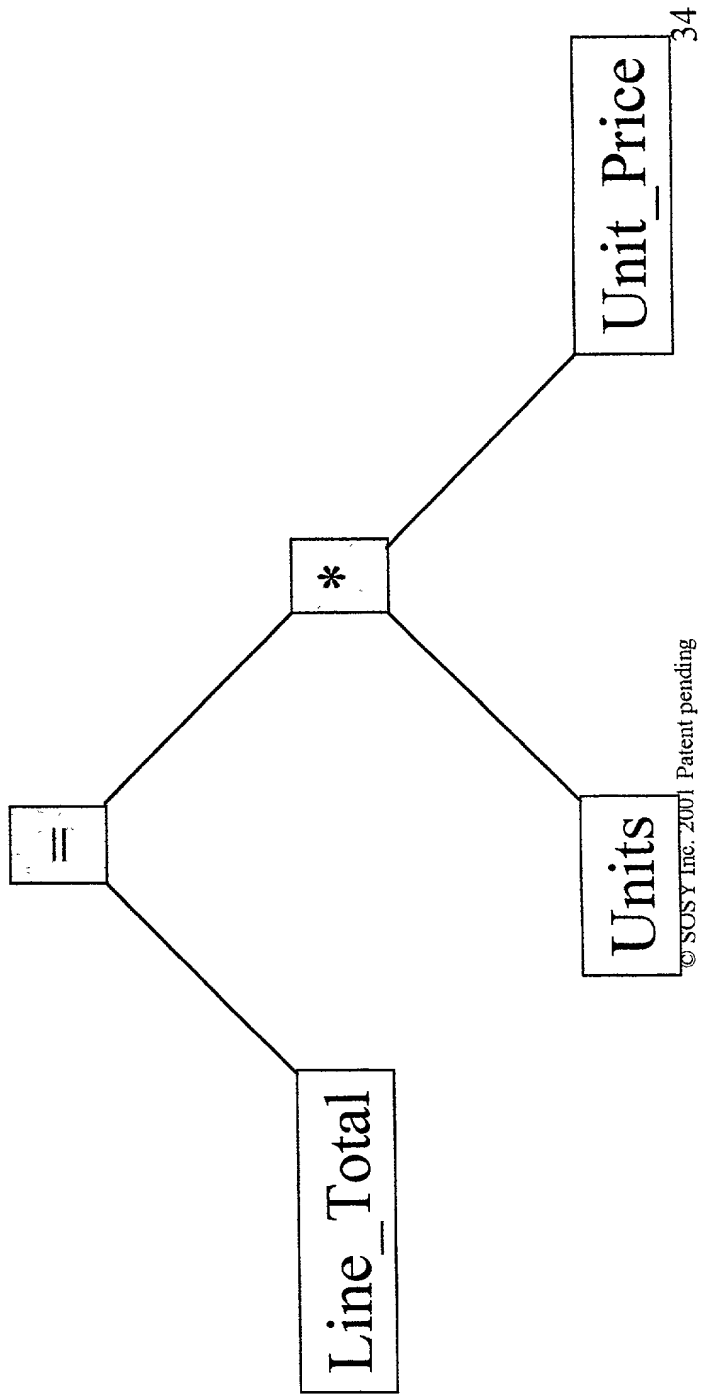
- Formulas of the Conceptual Model
 - Ensure the formulas of the Conceptual Model are correct and complete.
 - Syntactical and Semantical Validation according to an extended Formal Specification Language grammar.
- Input:
 - Formula expression
 - Formula Type (precondition, valuation, ...etc.)
 - Formula Context (class name, service name, ...etc.)
- Output:
 - Error Message (validation did not pass)
 - Validation Tree (validation ^{status}passed)

Validation Trees

- Binary Tree representation of a correct formula.
- Tree consists of Nodes and Leaves.
- Nodes
 - Represent operators
 - Can have one or two “branches” (binary)
 - Branches can again be nodes or leaves
- Leaves
 - Represent operands
 - Have no branches

Example

- $\text{Line_Total} = \text{Units} * \text{Unit_Price}$



Documentation Translation

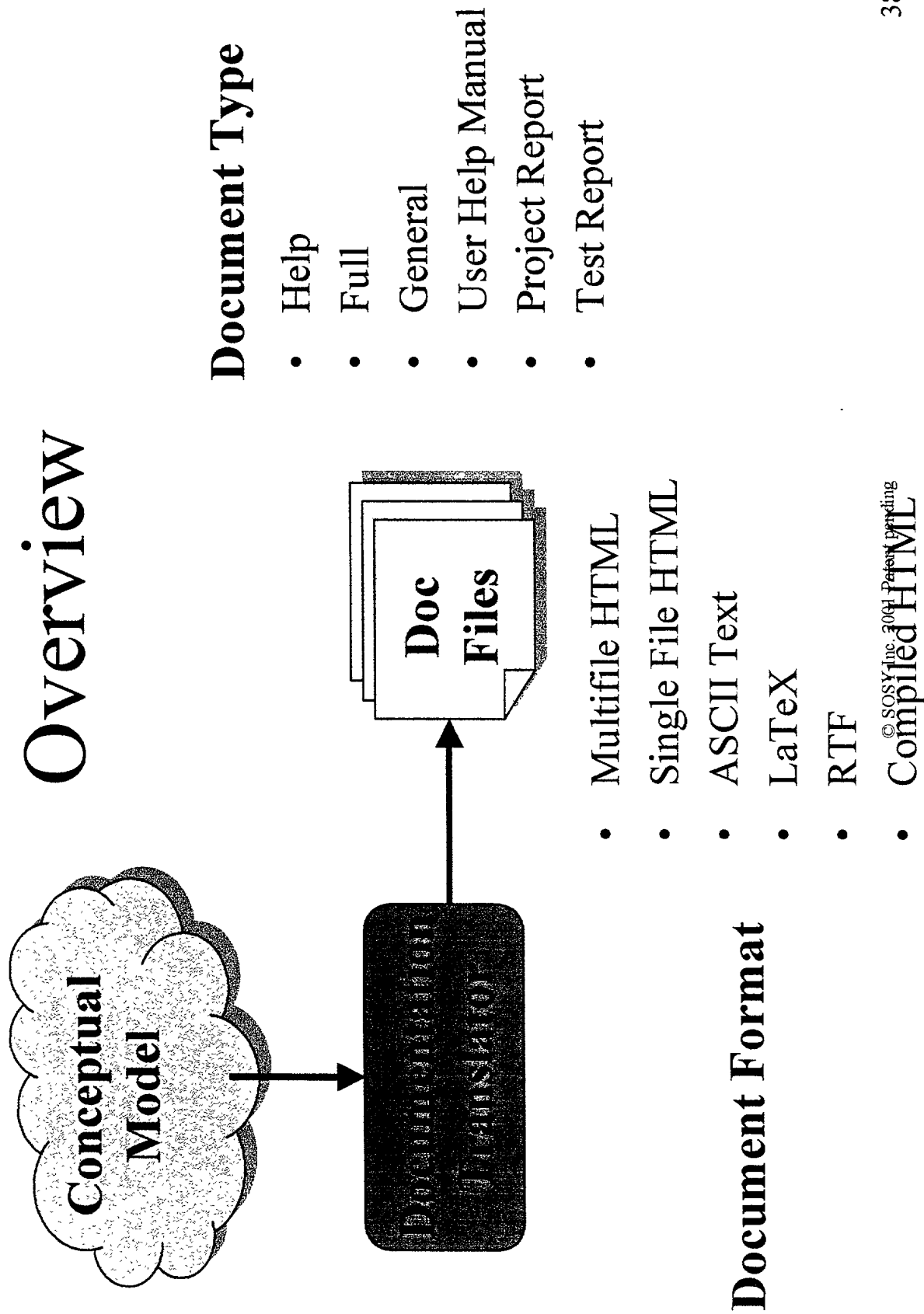
CARE Technologies, S.A.

Index

- Intro
- Overview
- Output Detail
 - Document Types
 - Document Formats
- Translation
 - CM Subset of Interest
 - Translation Process
 - Remarks
- Example

Intro

- Documentation Translation is the process to obtain, from a Conceptual Model, documentation on the system it represents.
- Documentation can have several degrees of detail and be focused on different aspects, thus obtaining different documentation formats from the same Conceptual Model.



Output Detail

- Document Types
 - Help
 - Description of each Class, its Attributes, Services and Population Selection Filters.
 - Full
 - Full description of a Conceptual Model
 - Aimed at analysts.
 - General
 - Description of each Class Attributes, Identification Function, Services, Aggregation Relationships and Specialization Relationships.

Output Detail

- Document Types
 - User Help Manual
 - Both Help Manual and Contextual Help (F1 key).
 - Intended for Operation Manual.
 - Integration with User Interface applications.
 - Project Report
 - Description of each Class Attributes and Services.
 - Test Report
 - Description of each Class Services.
 - Intended for Testing purposes.

Output Detail

- Document Formats
 - Multifile HTML
 - One HTML page per concept.
 - Recommended for navigable help.
 - Single File HTML
 - One single HTML page.
 - Recommended for printing.
 - ASCII Text
 - Single, plain ASCII text file.

Output Detail

- Document Formats
 - LaTeX
 - Single, LaTeX text file.
 - RTF
 - Single, RTF text file.
 - Compiled HTML
 - Same as Multifile HTML plus header files to be used by HTML Help Workshop compiler.
 - Recommended for contextual help.
 - Searching and Indexing facilities usage from browsers.

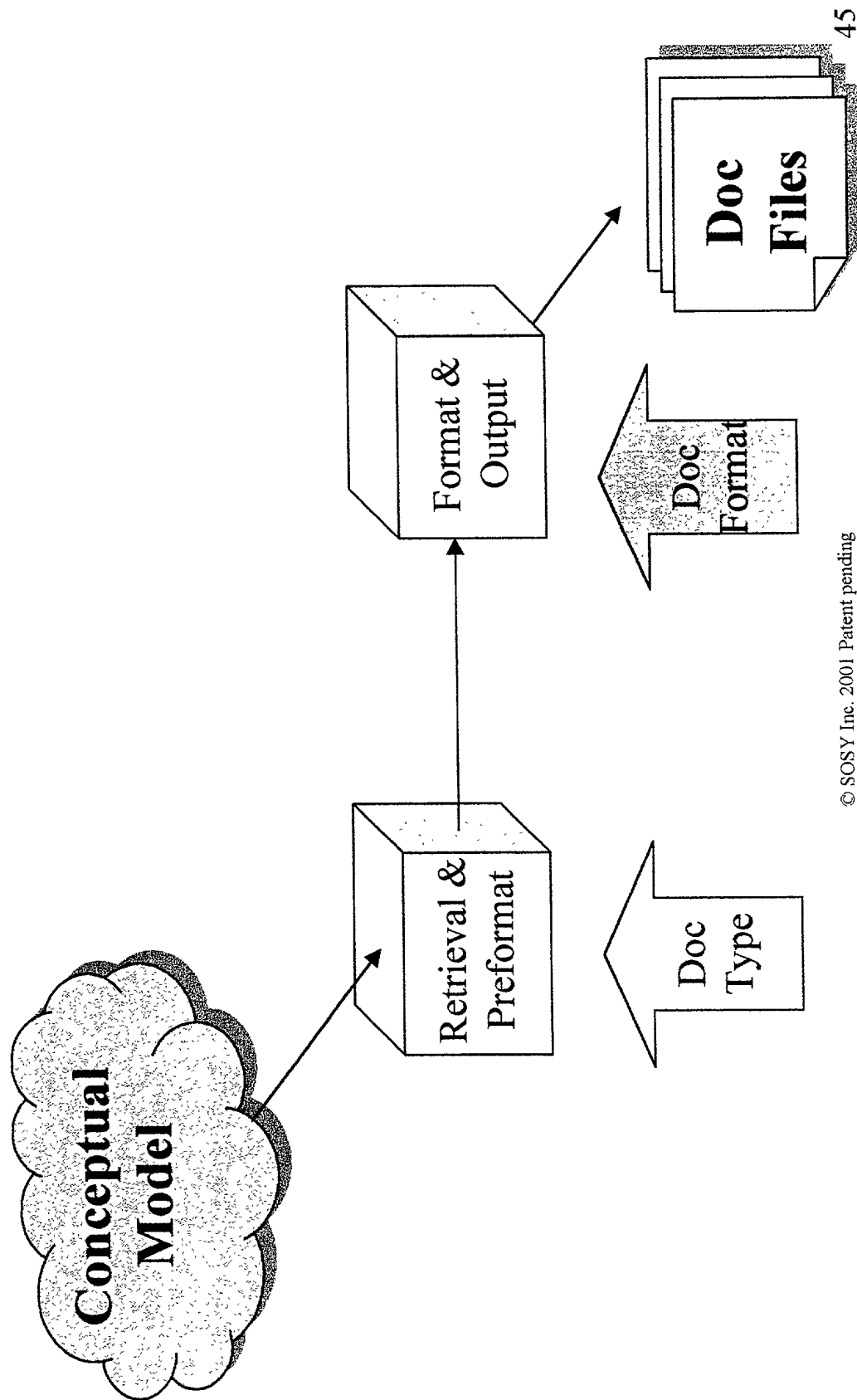
Translation

- Conceptual Model Subset of Interest
 - Subset of Interest depends on Document Type.
 - Usual elements:
 - Classes
 - Attributes
 - Relationships
 - Services & Arguments
 - Intensive use of analysis information.

Translation

- Translation Process
 - Read information from Conceptual Model and format it for output.
 - Two phases:
 - Information retrieval and pre-formatting.
 - Depends on Document Type
 - Independent from Document Format
 - Information output.
 - Depends on Document Format.
 - Independent from Document Type.

Translation Phases



Translation

- Remarks
 - Conceptual Model needs not to be valid (in terms of completeness and correctness) but it is always non-ambiguous.
 - The richer the analysis information, the richer the documentation.
 - Easily extensible
 - New Document Types
 - New Document Formats

Example

The screenshot shows a web browser window with the following elements:

- Address Bar:** [http://www.intranet.local/ExpenseReport/Doc Gen/Completa HTML/Expastos 1.0.html](#)
- Menu Bar:** Archivo, Edición, Ver, Favoritos, Herramientas, Ayuda
- Toolbar:** Includes icons for Back, Forward, Home, Stop, Reload, Print, and other standard browser functions.
- Page Content:**
 - Clase: Expense**
 - Propiedades:**
 - Alias:** Expense
 - Observaciones:** Employees may present a expense report when they have supported expenses on behalf of the company. (Agr Rel with Employee) Typically, the expenses are associated to a certain project or specific task. (Agr Rel with Project) At presenting the expense report, associated tickets will be attached out and advances will be reflected. Advances must be discounted out from the expense report balance. The expense report, once presented, must be authorized by a responsible of the expenses. The authorization process will allow reject the expenses if proceed. Once authorized, the expense report will be approved for payment by a responsible of accounting. Once paid, it will be marked as so. (Agr Status)
 - Mensaje de Ayuda:**
 - Hereda de:**
 - Se especializa en:**
 - Se relaciona con:** [Employee](#), [PaymentType](#), [Project](#), [MyCurrency](#), [Lines](#)
- Status Bar:** Intranet local

Persistence Relational Database Translation CARE Technologies, S.A.

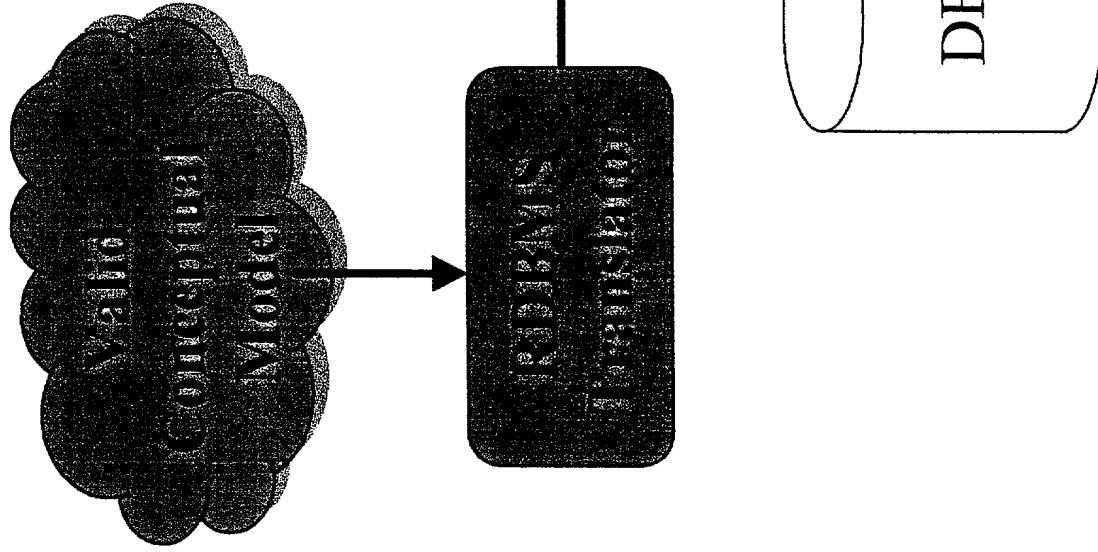
Index

- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Persistence Relational Database Translation is the process of creating a Relational Database from a certain subset of information in the Object Model of a valid Conceptual Model.
- Output script files are used to create a relational database using structured query language (SQL).

Overview



- Creates
- Primary Keys
- Foreign Keys
- Indexes
- Drop Creates
- Drop Primary Keys
- Drop Foreign Keys
- Drop Indexes

Output Detail

- **Creates**
 - Creation of Tables and Fields
- **Primary Keys**
 - Creation of Primary Keys as constraints on each table
- **Foreign Keys**
 - Creation of Foreign Keys as constraints on each table
- **Indexes**
 - Creation of Indexed on each table

Output Detail

- Drop Creates
 - Deletion of Tables
- Drop Primary Keys
 - Deletion of Primary Key Constraints
- Drop Foreign Keys
 - Deletion of Foreign Key Constraints
- Drop Indexes
 - Deletion of Indexes

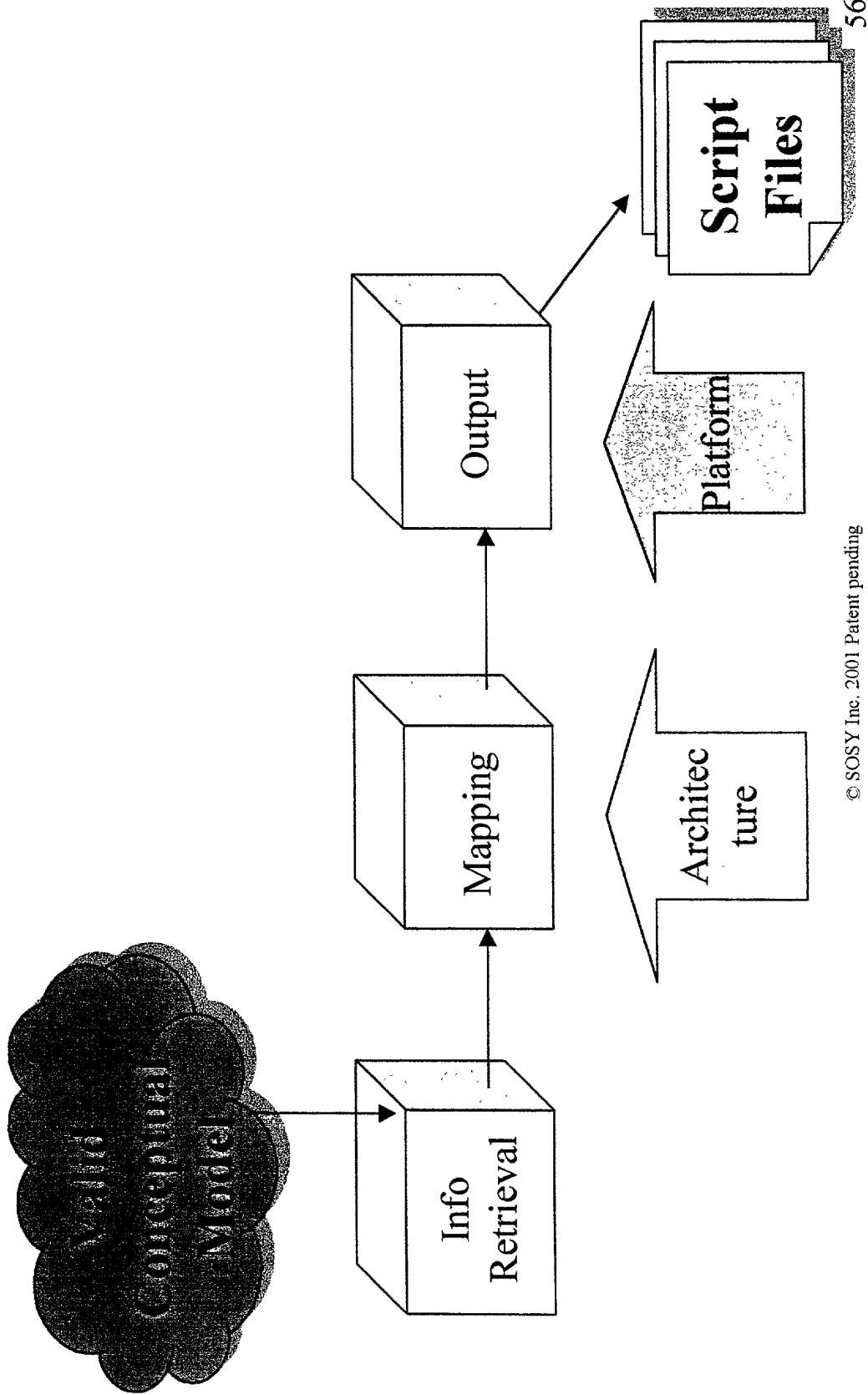
Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Classes
 - Attributes
 - Identification Functions
 - Aggregation Relationships
 - Inheritance Relationships

Translation

- Three phases:
 - Information retrieval.
 - Independent from persistence architecture.
 - Fixed architecture mapping.
 - Depends on persistence architecture.
 - Information output.
 - Targeted for Standard ANSI SQL 92 RDBMS.
 - Script files depends on the platform's SQL syntax of RDBMS manufacturer.
 - May depend on platform specifications to make use of manufacturer extensions and tuning.

Translation Phases



Translation

- Translation Processes. Mapping:
 - Class \rightarrow Table
 - Non-derived Attribute \rightarrow Field
 - Identification Function \rightarrow Primary Key
 - Univaluated Relationship \rightarrow Foreign Key
 - Univaluated Relationship \rightarrow Index
 - Multivaluated Relationship \rightarrow Table
 - Inheritance Relationship \rightarrow Foreign Key

Example

Create table script in SQL for Expense class

```
CREATE TABLE Expense (
    fk_Project_1 int NOT NULL ,
    id_Expense int NOT NULL ,
    fk_Employee_1 CHAR(10) NOT NULL ,
    fk_MyCurrency_1 CHAR(5) NOT NULL ,
    fk_PaymentType_1 CHAR(5) NULL ,
    PresentDate datetime NOT NULL ,
    Status int NOT NULL ,
    Cause VARCHAR(255) NOT NULL ,
    AuthoDate datetime NULL ,
    AuthoComments VARCHAR(255) NULL ,
    PaymentDate datetime NULL ,
    PayComments VARCHAR(255) NULL ,
    Advances DECIMAL(19,6) NOT NULL ,
    Exchange DECIMAL(19,6) NOT NULL);
```

Business Logic Translation

CARE Technologies, S.A.

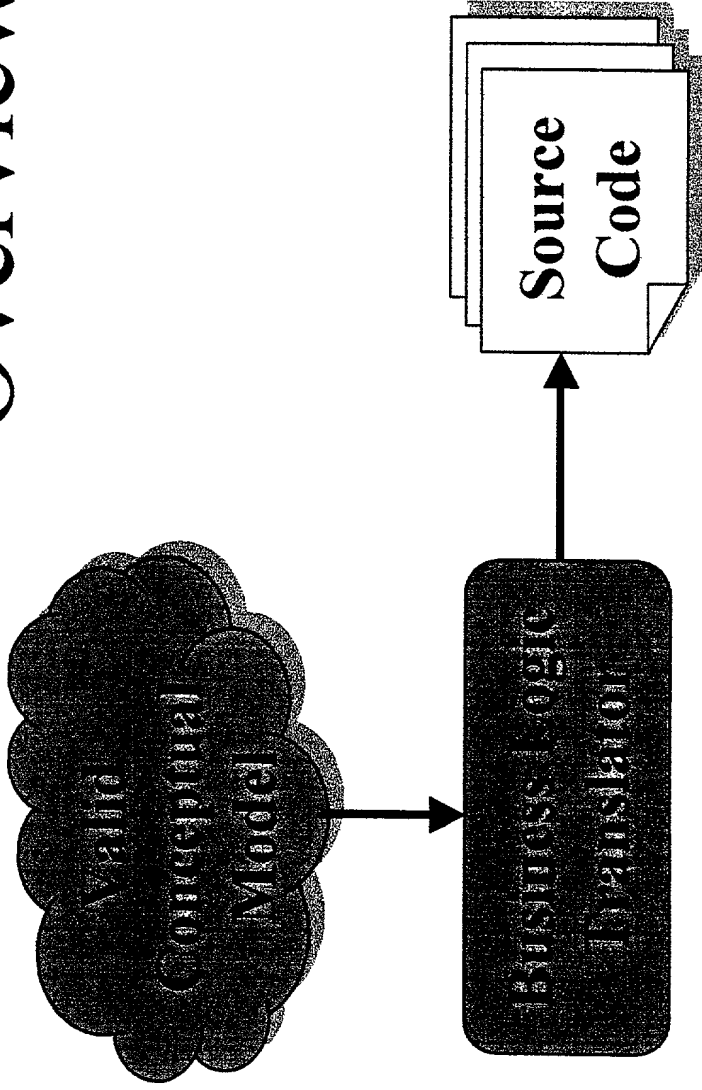
Index

- Intro
- Overview
- Output Detail
- Translation
 - CM Subset of Interest
 - Translation Processes
- Example

Intro

- Business Logic Translation is the process to obtain, following a precise Execution Model, the source code corresponding to the business logic from a valid Conceptual Model for a target Programming Language and Software Architecture.
- Execution Model is independent from Programming Language and Software Architecture.

Overview



Determines:

-Target Programming Language

-Target Software Architecture

Output Detail

- Target Programming Language and Software Architecture determine:
 - Source code organization in files
 - Files internal organization
- Source Code's backbone: Execution Model.

Output Detail

- Traceability: Source code highly readable and maintainable thanks to:
 - Source code is always organized and structured in the same way.
 - Naming conventions applied.
 - Source code includes analysis information from the Conceptual Model as comments.

Output Detail

- Implementation of a precise Execution Model grants Functional Equivalence with Conceptual Model.
- Programming Interface to Clients for:
 - Actor Validation and Authentication.
 - Services Execution.
 - Queries Execution.
- Manages:
 - Concurrency.
 - Transactions.
 - Interoperable Objects Persistence.

Translation

- Conceptual Model Subset of Interest
 - Object Model
 - Static properties (Visibility & Persistence)
 - Attributes + Identification Functions
 - Derivations
 - Aggregation Relationships
 - Inheritance Relationships
 - Services (Execution Model)
 - Arguments
 - Preconditions
 - Transaction Formulas
 - Actors (Execution Model)
 - Integrity Constraints (Execution Model)

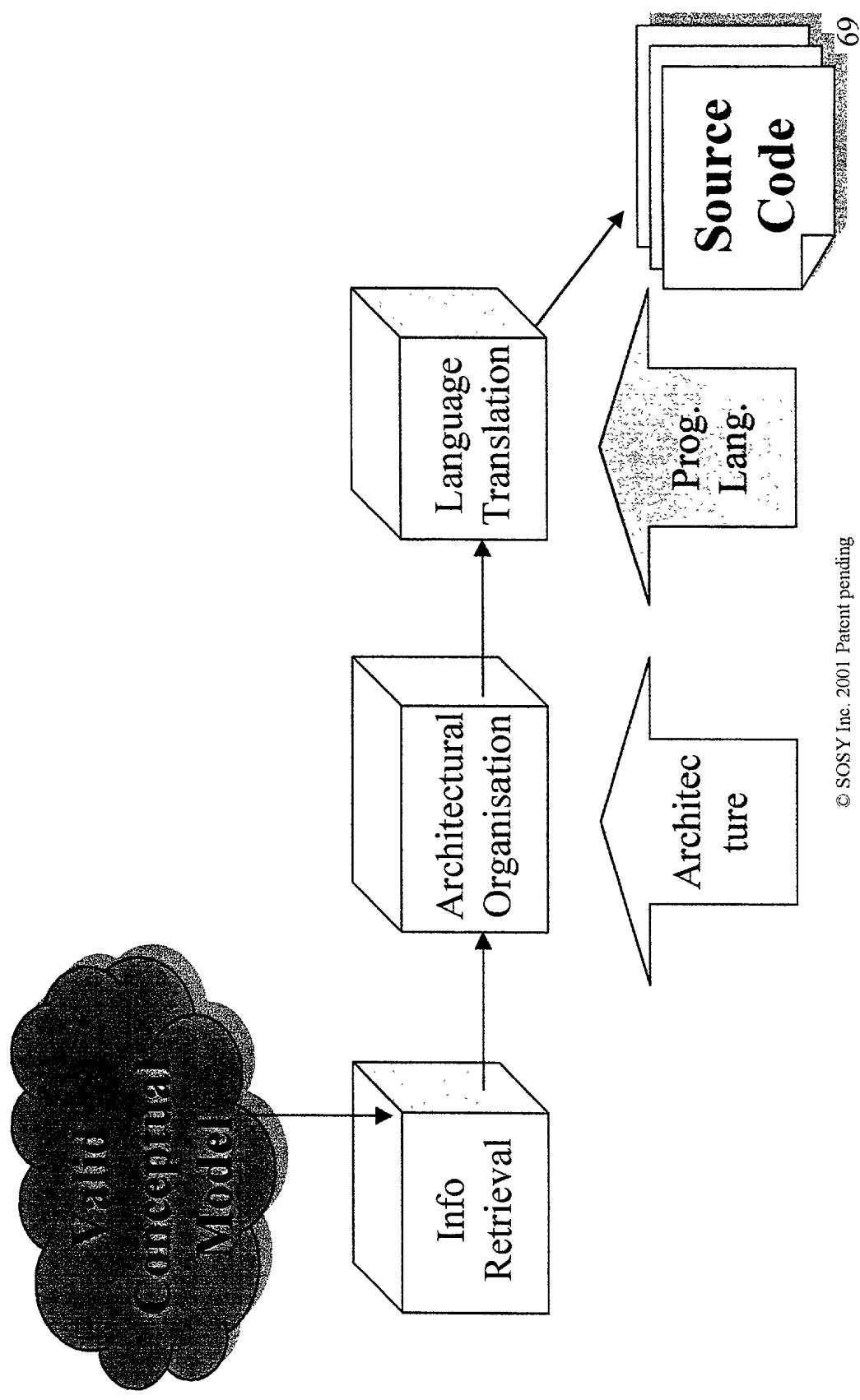
Translation

- Conceptual Model Subset of Interest.
 - Dynamic Model.
 - State Transition Diagram (Execution Model).
 - Controls Valid Lifes for an Object.
 - Object Interaction Diagram.
 - Triggers (Execution Model).
 - Global Transactions (Execution Model).
 - Functional Model (Execution Model).
 - Object state change upon occurrence of an event.

Translation

- Translation phases:
 - Information retrieval
 - Independent from target Software Architecture and Programming Language
 - Architectural organisation
 - Depends on target Software Architecture
 - Independent from target Programming Language
 - Determines files organisation and files internal structure
 - Language translation
 - Depends on target Programming Language
 - Influenced by Software Architecture
 - Takes advantage of Programming Language capabilities

Translation Phases



Translation

- Translation Processes
 - Classes
 - Static properties translation
 - Services translation
 - Queries translation
 - Global Interactions
 - Services translation
 - Global Functions
 - Functions Interface translation
 - Body is left blank

Example

- Evaluation:
 - Service Authorize modifies attributes Status, AuthoDate and AuthoComments
 - Formal Specification Language expression for evaluation Valuation
- [authorize ()] Status=2 and AuthoDate=today() and AuthoComments="";

- Visual Basic Produced

```
Private Function MV_kval_expense_authorize() As String
    Expense_Status = 2
    Expense_AuthoDate = today()
    Expense_AuthoComments = ""
    MV_kval_expense_authorize = ""
End Function
```

User Interface Translation

CARE Technologies, S.A.